SBPDU Frame	Recipients responsible for processing the frame	Number of copies for each recipient
DVMyInfo	$n' \in N'_B(n)$	1
DVOurInfo	$n' \in N'_B(n)$ s.t. $nca(n, n') = n$	1
DVInform	$n' \in N'_B(n)$ s.t. $nca(n, n') = n$	$\mid \{ k \mid k \in \{ N'_B(n')$ s.t. k and n' are on different branches $\} \mid$
	$n' \in N'_B(n)$ s.t. $nca(n, n') \neq n$	1 when the accurate distance is calculated by n but not n'; 0 otherwise
DVRecord	$n' \in N'_B(n) \cup N_B(n)$	At most $\|B\|$ for each $k \in B \backslash \{n, n'\}$
StationLoc	$n' \in B \backslash \{n\}$	$\mid \{ m \mid m \in M$ s.t. $ab(m) = n \} \mid$

Table 6: Summary of SBPDU Frames used in the STAR processes

Location information is necessary in all algorithms in Section 3 that are applicable for any additive metric. In those algorithms, every bridge has to know the location of all end stations. In the STAR Bridge Protocol, a STAR bridge keeps only the location of end station s provided $ab(s)$ is defined. Therefore, the location messages generated by the STAR Bridge Protocol are less than those generated by the algorithms in the prior art.

VII.C Path Length

In this section, we would like to show that the length of a STAR forwarding path is always less than or equal to the corresponding tree path. In the following discussion, we denote the length of the STAR forwarding path between two bridges x and y as $len(x, y)$. In all figures referring to this section, a black node represents a STAR bridge, a white node represents an old bridge, and a dot-dash line represents a tree path. We first establish the following lemmas.

Lemma 1:

When a STAR bridge n receives a normal data frame originated by an end station s, the STAR bridge may encapsulate the frame only if $n = ab(s)$.

Proof of Lemma 1:

We observe that encapsulation is executed only in the ESL_Search_Proc procedure (Pseudocode 8) and the DF_STAR_Forwarding_Proc procedure (Pseudocode 9) in which the ESL_Search_Proc procedure is invoked. In DF_STAR_Forwarding_Proc procedure, when n receives a normal data frame,

the ESL_Search_Proc procedure is invoked by *n* only when *n* = *ab(s)* (Case 9.2b). Note that if *ab(s)* is not defined, *n* can never be *ab(s)*.

Lemma 2:

A normal data frame originated by an end station *s* will always be forwarded over a tree path if the frame is not encapsulated by *ab(s)*.

Proof of Lemma 2:

According to Case 8.3b in the ESL_Search_Proc procedure, the normal data frame is never forwarded over a crosslink, unless it is encapsulated. By Lemma 1, if the normal data frame is not encapsulated by *ab(s)*, it will remain as a normal data frame for the rest of its forwarding journey. According to Case 9.2a and Case 9.2c of the DF_STAR_Forwarding_Proc procedure, having received the normal data frame, a STAR bridge *n* that is not *ab(s)* may forward the frame only to tree neighbors. Having received the normal data frame, an old bridge may forward the frame only to tree neighbors. Therefore, the normal data frame will be forwarded over a tree path if the frame is not encapsulated by *ab(s)*.

Lemma 3:

If a frame is forwarded from an end station *s* to another end station *t* over an enhanced forwarding path, the path must traverse at least one crosslink, and the frame must be encapsulated by *ab(s)*, which is necessarily defined.

Proof of Lemma 3:

If *s* and *t* are on the same branch, then the tree path from *s* to *t* is necessarily a shortest path. Given that *s* and *t* are on different branches, the forwarding may traverse two or more different branches. In the first case, the forwarding path must traverse a crosslink, or it would not be an enhanced forwarding path. In the second case, it must traverse at least one crosslink because the forwarding path cannot be a normal tree path. According to Case 8.3b in the ESL_Search_Proc procedure, the frame will be encapsulated when it is forwarded over any crosslink. By Lemma 2, the frame must be encapsulated by *ab(s)*.

Lemma 4:

If a STAR forwarding path, along which a frame is sent from an end station *s* to another end station *t*, is an enhanced forwarding path, then *ab(s)* must be defined and is the first STAR bridge on the enhanced forwarding path.

Proof of Lemma 4:

By Lemma 3, the frame must be encapsulated by *ab(s)*, which is necessarily defined. Since *ab(s)* is by definition the first STAR bridge on the root path of *db(s)*. If *ab(s)* = *db(s)*, then the proof is complete. Otherwise, *db(s)* is an old bridge, and will send a copy of the frame along the root path of *db(s)* regardless of its knowledge of the end station *t*. This copy of the frame will be received *ab(s)*, which will encapsulate the frame so that it will be forwarded over the enhanced forwarding path. Duplicate copies of the frame will be dropped in accordance with the protocol.

Lemma 5:

If a STAR forwarding path, along which a frame is sent from an end station *s* to another end station *t*, is an enhanced forwarding path, then *ab(t)* must be defined and is the last STAR bridge on the enhanced forwarding path.

Proof of Lemma 5:

By Lemma 3, the forwarding path must traverse at least one crosslink. According to Case 8.3b in the ESL_Search_Proc procedure, the frame will be encapsulated when it is forwarded over each crosslink. *ab(t)* must be defined because, according to Case 9.2a of DF_STAR_Forwarding_Proc procedure, a frame will not be encapsulated otherwise. If *ab(t)* = *db(t)*, the proof is complete. Otherwise, *ab(t)* must be an ancestor of *db(t)* by definition, and *db(t)* must receive the frame without encapsulation. It suffices to show that *ab(t)* is on the enhanced forwarding path and all intermediate STAR bridges on the enhanced forwarding path, except *ab(t)*, will forward the frame with encapsulation. *ab(t)* is on the enhanced forwarding path because, according to Case 8.3b of the ESL_Search_Proc procedure, each intermediate STAR bridge forwards the encapsulated frame through its forwarding port leading to *ab(t)*. According to Case 8.2 and Case 8.3 in the ESL_Search_Proc procedure, a STAR bridge will forward the frame without encapsulation only if it is *ab(t)*.

Lemma 6:

When *ab(s)* and *ab(t)* are both defined, and they are on different branches of the spanning tree, $len(db(s), db(t)) \leq d_T(db(s), db(t))$.

Proof of Lemma 6:

FIGs. 23a-23c respectively show various exemplary scenarios for this lemma. Since *ab(s)* and *ab(t)* are on different branches, *s* and *t* must be on different branches. If the forwarding path from *db(s)* to *db(t)* is a tree path (Fig 23a), then the proof is complete. Otherwise, the forwarding path is an enhanced forwarding path. By Lemma 4, *ab(s)* is the first STAR bridge on the enhanced forwarding path. By